**UNIT 1**

**Logic Families and XOR-XNOR Gate applications:**

**Logic Specifications:**

The specification sheets for various logic families contain many terms unique to digital electronics. Some of the important specifications of logic families are listed below.

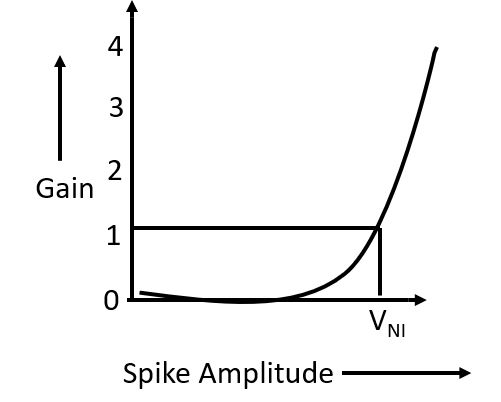
1. Fan in
2. Fan out
3. Noise immunity and Noise margin
4. Propagation delay

**Fan in:** The fan-in of a logic circuit is the number of inputs that logic circuit can handle. For example, an eight-input gate requires one unit load per input. Therefore, its fan in is eight.

**Fan out:** The fan out of a circuit is the number of unit inputs that can be driven by a logic element. If the gate has fan out of six, this means it can drive six-unit inputs and still maintain its logical 1 and logical 0 output voltage specifications.

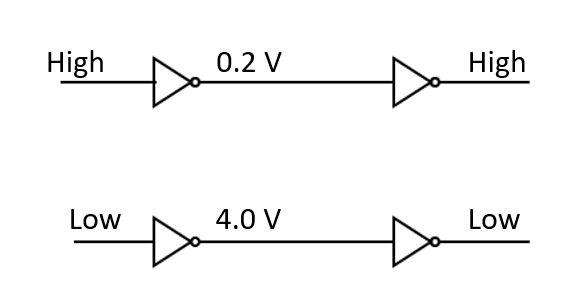
**Noise Immunity and Noise Margin:** Noise immunity represents the amount of noise a logic circuit can handle without being amplified beyond unity gain.

There is some spike amplitude at which the gain of the device is unity (1). This amplitude is called the device immunity, VNI.



**FIG 1: A graph of Gain versus Spike Amplitude**

**Noise Margin:** A second noise specification is noise margin. Assume a circuit consist of two invertors.



**FIG 2: NOT gate showing Output voltages**

According to manufacturer, the highest logical 0 any invertor will ever output is 0.2 V. However, the manufacturer also guarantees that any input less than 1.0 V will always be considered a logical 0. Therefore, we can say 1.0V-0.2V=0.8V of noise on the input to a second stage and still call it logical 0. This 0.8 V is called “noise margin VNM”.

**Mathematically VNML=VILMax -VOLMax**

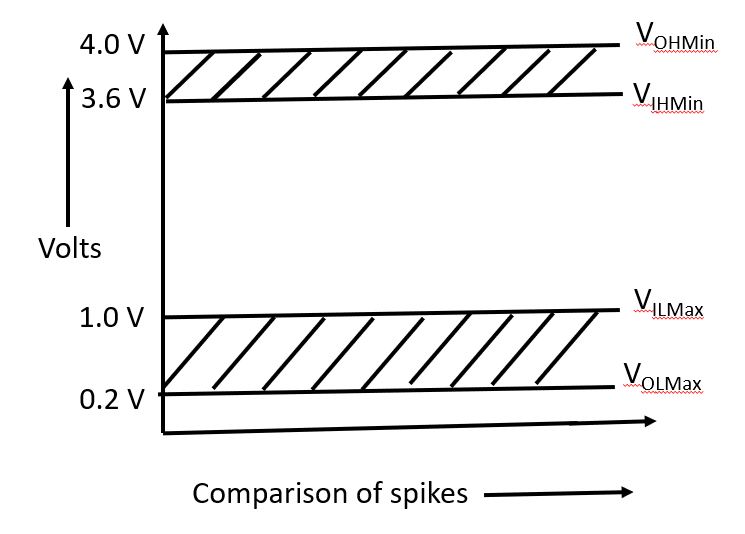
The same system of reasoning applies to high level noise margin, VNMH. If the manufacturer guarantees the minimum logical 1 at a gate output to be 4.0 V but states that any voltage down to 3.6 V is to be considered a logical 1 input,

The noise margin is

**VNMH = VOHMin -VIHMin**

= 4.0 V-3.6 V =0.4 V

Graphically it can be shown as

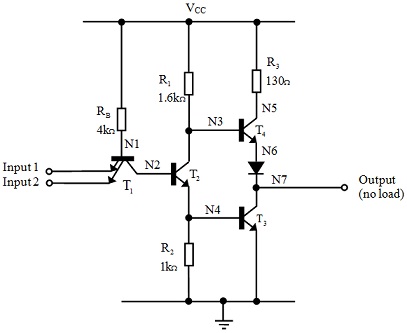


**FIG 3: Figure shows Noise Margin for Low and High State**

**Propagation Delay:** It is time taken by the pulse to get through a logic device measured at 50 % of the peak. This is an important consideration for it determines how fast a logic circuit can operate.

**Transistor Transistor Logic (TTL) Circuit for NAND gate:**

**FIG 4: Transistor Transistor Logic (TTL) Circuit for NAND gate:**



TTL is called T2L family and was introduced to provide greater speed than Diode Transistor Logic (DTL). A 74 series TTL NAND gate is shown in the figure 4 and truth table of NAND gate in Table 1.

|  |  |  |
| --- | --- | --- |
| A | B | Y= |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**TABLE 1: Truth Table of NAND Gate**

If both the inputs A and B are high, T1 has no emitter current. However, its base collector junction is forward biased supplying base current to T4 causing it to conduct. Thus, the output at collector of T3 will go low at low potential. (For A=B=1, Output=0). However, T2 collector will go low and hence no base current to T3and so T3 will go to cut off.

If either A or B or both A and B inputs are low, then T1 will have base emitter current saturating it and pulling T2 base to ground and cutting it off. This will cause T3 to conduct and T4 to cut off.

Note that this TTL family conducts when ON or OFF. This means that the output line will always have a low impedance reducing the effect of noise. Unused inputs on TTL gate are subject to noise and should be connected to a defined logic level. They may be connected to either a used input or through a resistor to Vcc. These resistors are called pull up resistors and make sure that base -emitter junction of the input transistor is reverse biased.

The logic specifications for TTL gates are as follows.

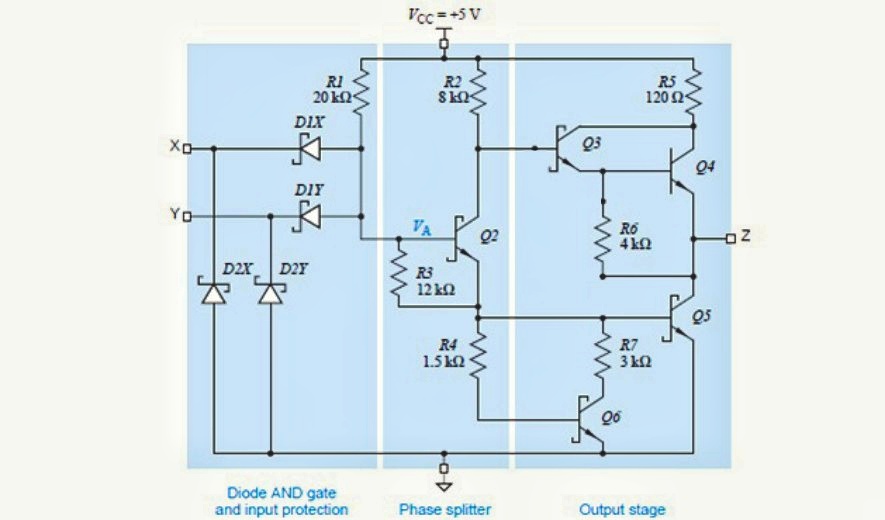
|  |  |
| --- | --- |
| Power Supply Vcc | 5 V |
| Power dissipation Pd | 100 mW |
| Propagation Delay Td | 15 nsec |
| Noise Margin VNM | 0.4 V |
| FAN Out FO | 10 |

**Table 2: Logic specifications for TTL NAND gate.**

The 74LS (Low Schottky) series has both higher speed and lower power dissipation compared to the 74 LS series. The circuit diagram of Low Schottky TTL NAND gate is shown in figure 5 and logic specifications for 74LS TTL NAND gate are shown in Table 3.

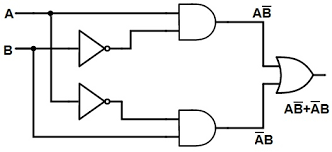
|  |  |
| --- | --- |
| Power Supply Vcc | 5 V |
| Power dissipation Pd | 22 mW |
| Propagation Delay Td | 10 nsec |
| Noise Margin VNM | 0.3 V |
| FAN Out FO | 20 |

**Table 3: Logic specifications for LS74 TTL NAND gate.**



**FIG 5: Schottky TTL Circuit for NAND gate**

**Applications of XOR and XNOR gates:**

XOR Gate: Figure 6 shows circuit diagram of Exclusive OR gate. It has two inputs and one output. Each input goes into an inverter. The outputs of the inverters are and . and B goes into upper AND gate and its output is B. Likewise A is the output of lower AND gate. The OR gate has inputs B and A. The final output of XOR gate thus becomes Y=B +A.

**FIG 6: Circuit diagram of XOR gate**

It is known as XOR gate because the truth table of XOR gate resembles the truth table of OR gate excluding only one condition A=B=1. It is represented by ⊕ symbol. The truth table and logic symbol of XOR gate are shown in figure 6 and Table 3.

|  |  |
| --- | --- |
|  | XOR Gate Truth Table |

**FIG 7: Logic symbol of XOR gate Table 4: Truth Table of XOR gate**

The output of XOR gate is high if and only if two inputs are different. XNOR gate is complement of XOR gate. The logic symbol and truth table of XNOR gate are shown in Figure 8 and Table 4 respectively.

|  |  |
| --- | --- |
| symbol of xnor gate | truth table xnor gate |

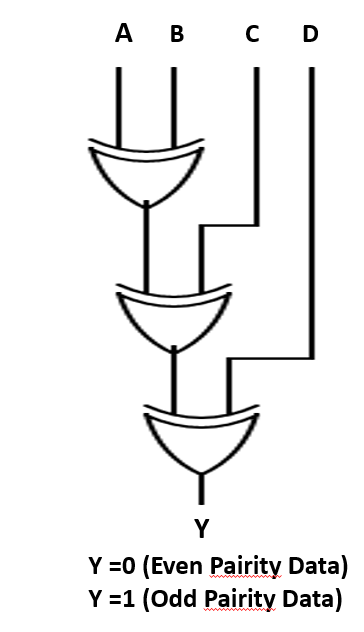
**FIG 8: Logic symbol of XNOR gate Table 5: Truth Table of XNOR gate**

The output of XNOR gate is high if and only if both the inputs are different.

**Applications of XOR gate:**

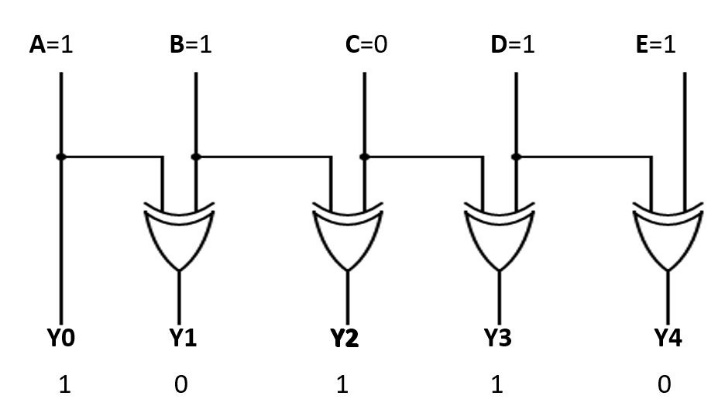
**Pairity Checker:** Pairity is an extra bit called to the data to make total number of 1’s either even or odd. If total number of 1’s is even it is called Even Pairity data and if total number of 1’s is odd it is called Odd Pairity data.

Figure 9 shows circuit diagram of three bit pairity checker with ABCD as data inputs. If ABCD = 1001, the output of first XOR gate is 1. So, input to second XOR gate is 1 and 0 So its output is also 1. Now, input to third XOR gate is 1 and 1 and hence its output is 0. So, it confirms that data is even pairity data. If ABCD is odd pairity data then the output will be Y=1. This pairity checker is used in telecommunication system to detect one bit error.



**FIG 10: Pairity checker**

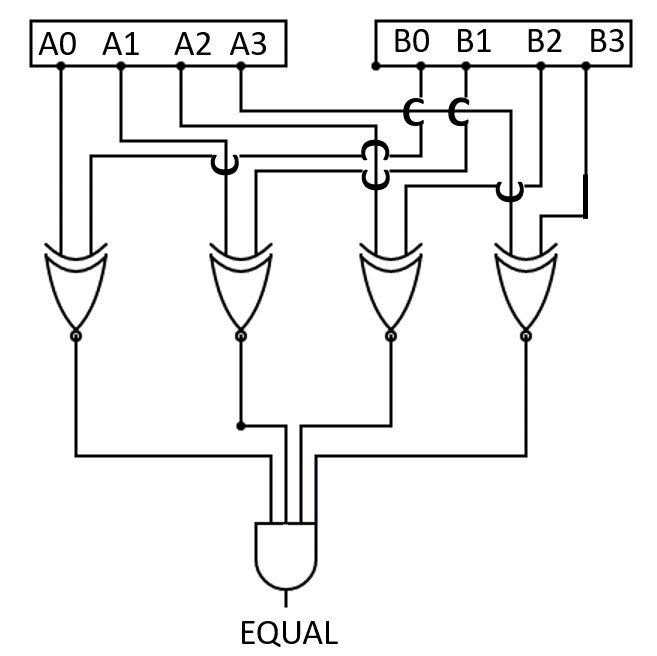
**Code Converter:** The code converter circuit is used to convert binary data to equivalent Gray code data. XOR gates are used to construct the code converter circuit. XOR gate performs Mod 2 addition. Figure 10 shows code converter circuit and the inputs are labelled as A,B,C,D,E and Output as Y0,Y1,Y2,Y3,Y4. Let A,B,C,D, E =1 1 0 1 1 then the output will be 1 0 1 1 0. Any binary data can be converted to Gray code using given circuit. Note that number of XOR gates used depends on how many data bit code is to be converted. For e.g. for 5 bit data, it requires 5 XOR gates.



**FIG 11: Binary to Gray Converter**

**Comparator:**

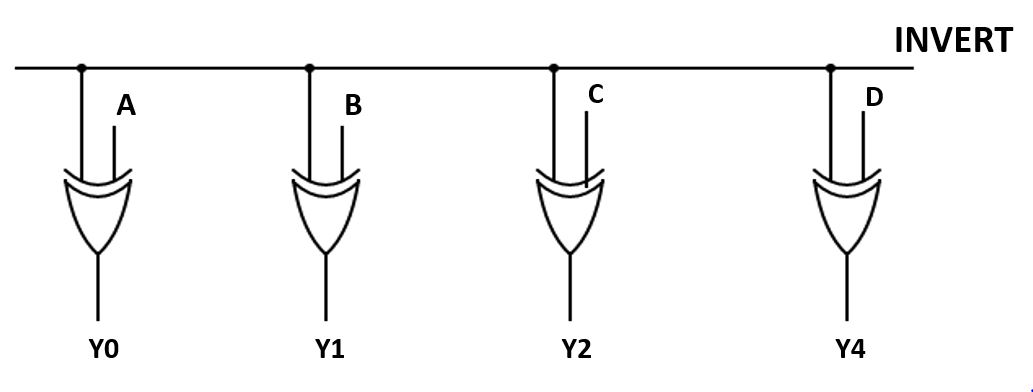
Comparator is a circuit which compares two binary words. If two binary words are equal then EQUAL =1. And if two words are different then EQUAl ≠ 1 but EQUAL =0. The circuit consists of four XNOR gates and one four input AND gate. We remember here that the output of XNOR gate will be high if and only if all inputs are same, otherwise if bits is different then output will be low. Here if A0 A1 A2 A3 = B0 B1 B2 B3 then all inputs of AND gate are equal and therefore its output will be 1. But if A0 A1 A2 A3 ≠ B0 B1 B2 B3 then output of AND gate will be 0 which reflects two data inputs are different.



**FIG 12: Four Bit Comparator**

**Controlled Inverter:**

Figure 13 shows circuit diagram for controlled inverter using XOR gates. When INVERT line is high, the data A B C D will be complemented. When INVERT line is low, the same data will be transmitted.



**FIG 13: Controlled Inverter**

For example, when A B C D = 1 0 1 0 and INVERT line is high, then first XOR gate has input 1 1, second XOR gate has input 0 1, third XOR gate has input 1 1 and fourth XOR gate has input 0 1. So, output of first XOR gate will be 0, output of second XOR gate will be 1, output of third XOR gate will be 1 and output of fourth XOR gate will be 1.

So, for A B C D =1 0 1 0 we have Y0 Y1 Y2 Y3 as 0 1 0 1, which is complement of input data. Similarly, if INVERT line is low, for A B C D =1 0 1 0 we will have Y0 Y1 Y2 Y3 = 1 0 1 0, which means same data is transmitted. Controlled inverters are used in communication system.

**Half Adder and Full Adder:**

The half adder is a circuit that adds two binary bits at a time. Figure 14 shows how to construct half adder circuit and Table 6 represents truth table of Half adder circuit.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FIG 14: Half Adder Circuit** | |  |  |  |  | | --- | --- | --- | --- | | **A** | **B** | **SUM** | **CARRY** | | 0 | 0 | 0 | 0 | | 0 | 1 | 1 | 0 | | 1 | 0 | 1 | 0 | | 1 | 1 | 0 | 1 |   **TABLE 6: Truth Table of Half Adder** |

If we examine the output column, then SUM can be represented by A⊕B (XOR Gate) and CARRY can be represented by A●B i.e., AND gate. Half adder is an elementary circuit and it takes first step towards circuit capable of more difficult arithmetic.

**Full Adder:**

When adding two binary numbers, we may have a carry from one column to the next column. So, in the next column we must add three bits because of carry. To add binary numbers electronically, we need a circuit that can handle three bits at a time. By connecting two half adders and an OR gate we get a full adder, a circuit that can add three bits at a time. The truth table for three input full adder circuit is shown below in Table 7. **TABLE 7: Truth Table of Full Adder**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FIG 15: Block diagram of Full Adder Circuit** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | A | B | C | CARRY | SUM | | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 0 | 1 | | 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 1 | 1 | 0 | | 1 | 0 | 0 | 0 | 1 | | 1 | 0 | 1 | 1 | 0 | | 1 | 1 | 0 | 1 | 0 | | 1 | 1 | 1 | 1 | 1 | |

**Half and Full Subtractor:**

Half subtractor is a circuit that subtracts one binary from another. The truth table and circuit diagram of Half subtractor is shown in Table 8 and figure 16 respectively. Looking to the difference column, we find that the output is high when two inputs are different. Hence, it is XOR gate. But for borrow we don’t have specific logic gate. The output is high for condition Ā B only. Hence half subtractor circuit can be constructed as follows.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  | | --- | --- | --- | --- | | **A** | **B** | **SUM** | **CARRY** | | 0 | 0 | 0 | 0 | | 0 | 1 | 1 | 0 | | 1 | 0 | 1 | 0 | | 1 | 1 | 0 | 1 |   **TABLE 8: Truth Table of Half   Subtractor** |

The half subtractor handles only two bits at a time and can be used for the least significant column of the subtraction. Full subtractor is designed to take care of a higher order column. It is constructed using two half subtractors and an OR gate as shown in Fig. 17. The truth table for full subtractor is shown in Table 9.

**TABLE 9: Truth Table of Full Subtractor**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FIG 16: Block diagram of Full Subtractor Circuit** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | A | B | Input Borrow | Difference | Borrow  Output | | 0 | 0 | 0 | 0 | 0 | | 0 | 1 | 0 | 1 | 1 | | 0 | 0 | 0 | 1 | 0 | | 0 | 1 | 0 | 0 | 0 | | 1 | 0 | 1 | 1 | 1 | | 1 | 1 | 1 | 0 | 1 | | 1 | 0 | 1 | 0 | 0 | | 1 | 1 | 1 | 1 | 1 | |